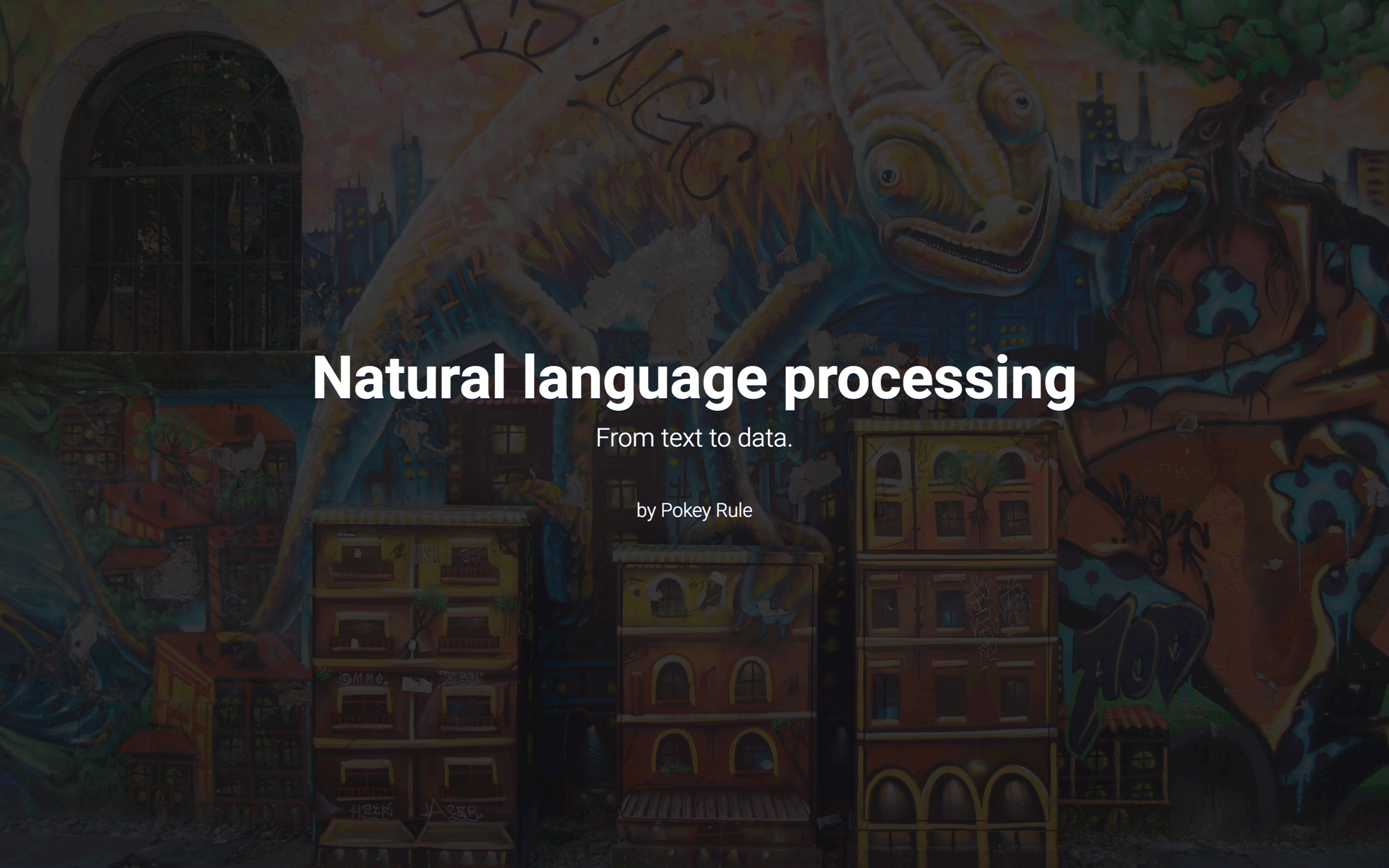


Natural language processing

From text to data.

by Pokey Rule

Table of contents

1. What is NLP?
2. NLP at Globality
3. Our NLP pipeline
4. Relation extraction
5. Our implementation of relation extraction

What is NLP?

> **Natural language processing (NLP)** is a field of computer science, artificial intelligence and computational linguistics concerned with the interactions between computers and human (natural) languages, and, in particular, concerned with programming computers to fruitfully process large natural language corpora.
>
> — *Wikipedia*

Areas of NLP

Syntax	**Semantics**	**Discourse**	**Speech**
Lemmatization	Machine translation	Automatic summarization	Speech recognition
Part-of-speech tagging	Named entity recognition (NER)	Coreference resolution	Speech segmentation
Parsing	Natural language generation	Discourse analysis	Text-to-speech
Sentence breaking	Natural language understanding		
Word segmentation	Question answering		
	Relation extraction		
	Sentiment analysis		
	Topic recognition		
	Word sense disambiguation		

Source: Wikipedia

NLP at Globality

Provider network growth

Automatically discover new service providers and infer their location, service type, and industry experience

Dynamic Q&A (Onboarding/Project Brief)

Infer information about a client or service provider based on free-form textual input

Conversational agents

Chat bots can automate common user interactions.

Pipeline

01. Tokenization

Split the text into individual tokens

```
["We", "are", "located", "in", "Germany"]
```

02. POS tagging

Determine the part of speech of each token

We	are	located	in	Germany.
PRP	VBP	VBN	IN	NNP

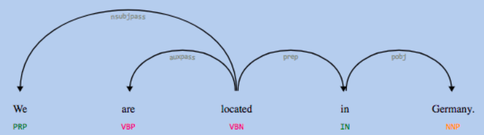
03. Named entity recognition

Find entities corresponding to nodes in knowledge graph

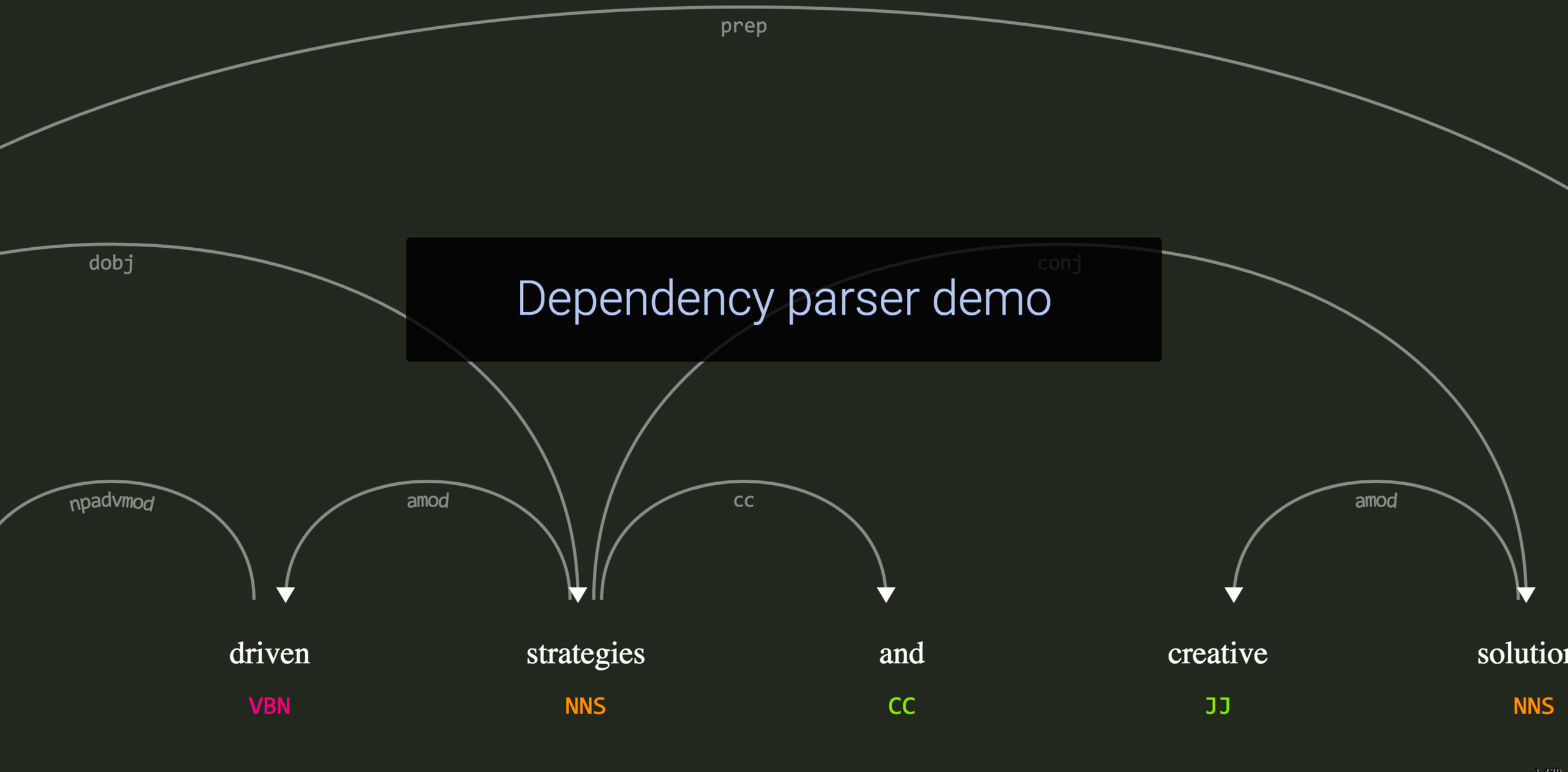
```
http://graph.globality.io/resource/Germany
```

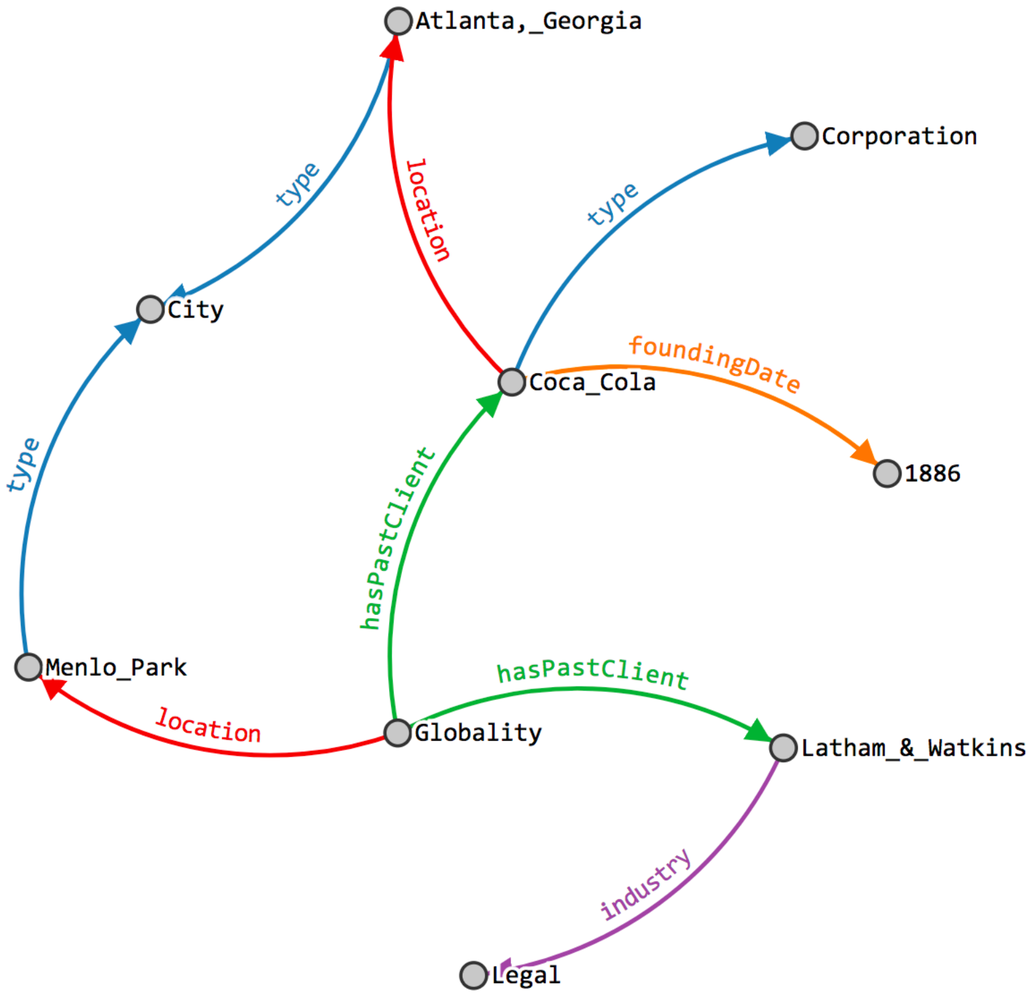
04. Dependency parsing

Parse the sentence into a hierarchical structure



Dependency parser demo



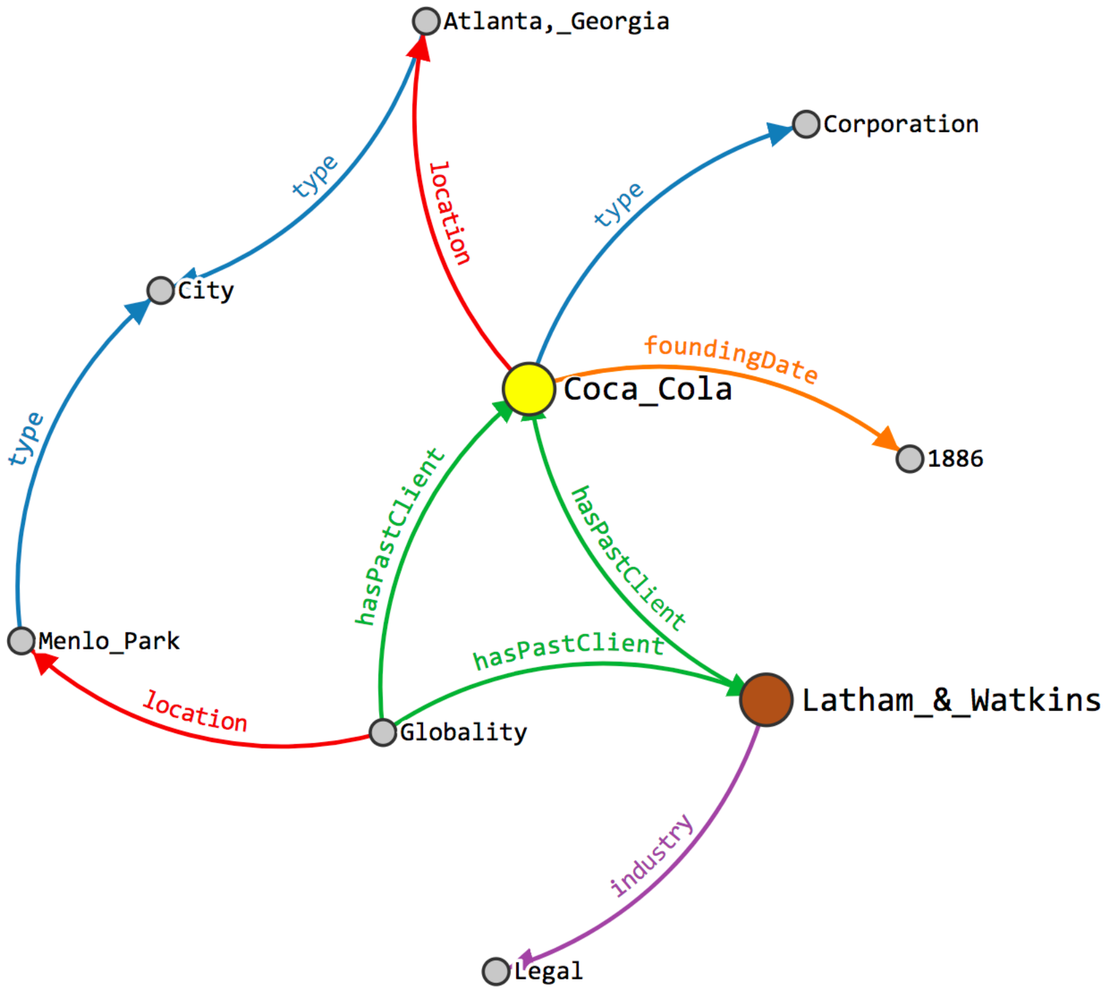


Knowledge graph

Knowledge graph is a collection of relations between entities.

Contains information useful for matching

Given a sentence, we'd like to find relevant entities in knowledge graph and add new relations

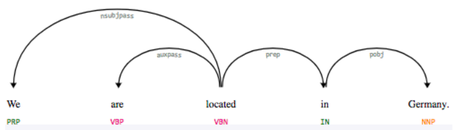


Relation extraction

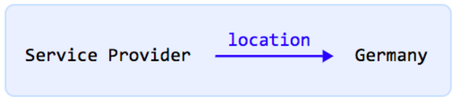
Latham & Watkins has had many high-profile clients, including Coca-Cola.

Using the dependency parse

Given a parsed sentence, we'd like to fit it into the knowledge graph




```
We are located in Germany.
```

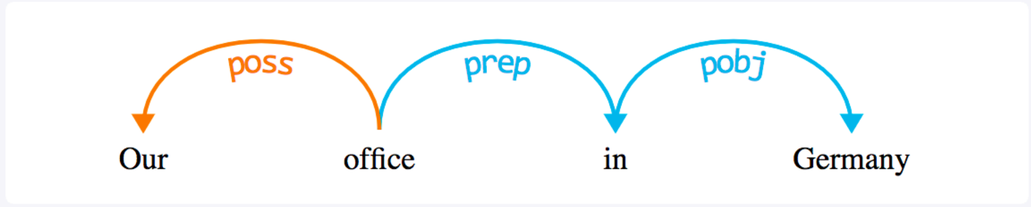


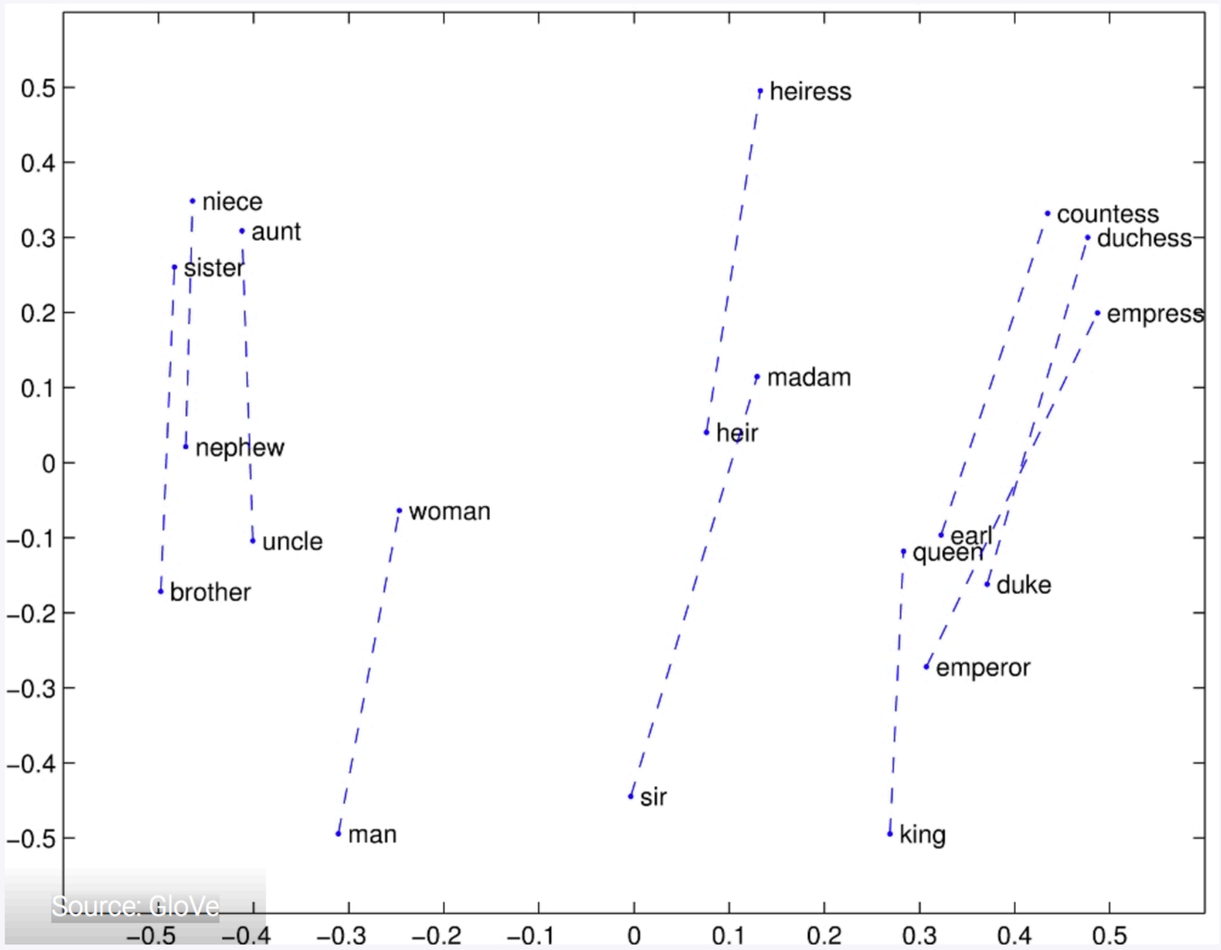
Dependency tree paths

Once text is parsed, we find path between entity and service provider

Desired sentences will have specific grammatical patterns

We need a way to describe the set of grammatical patterns that are indicative of desired relationship





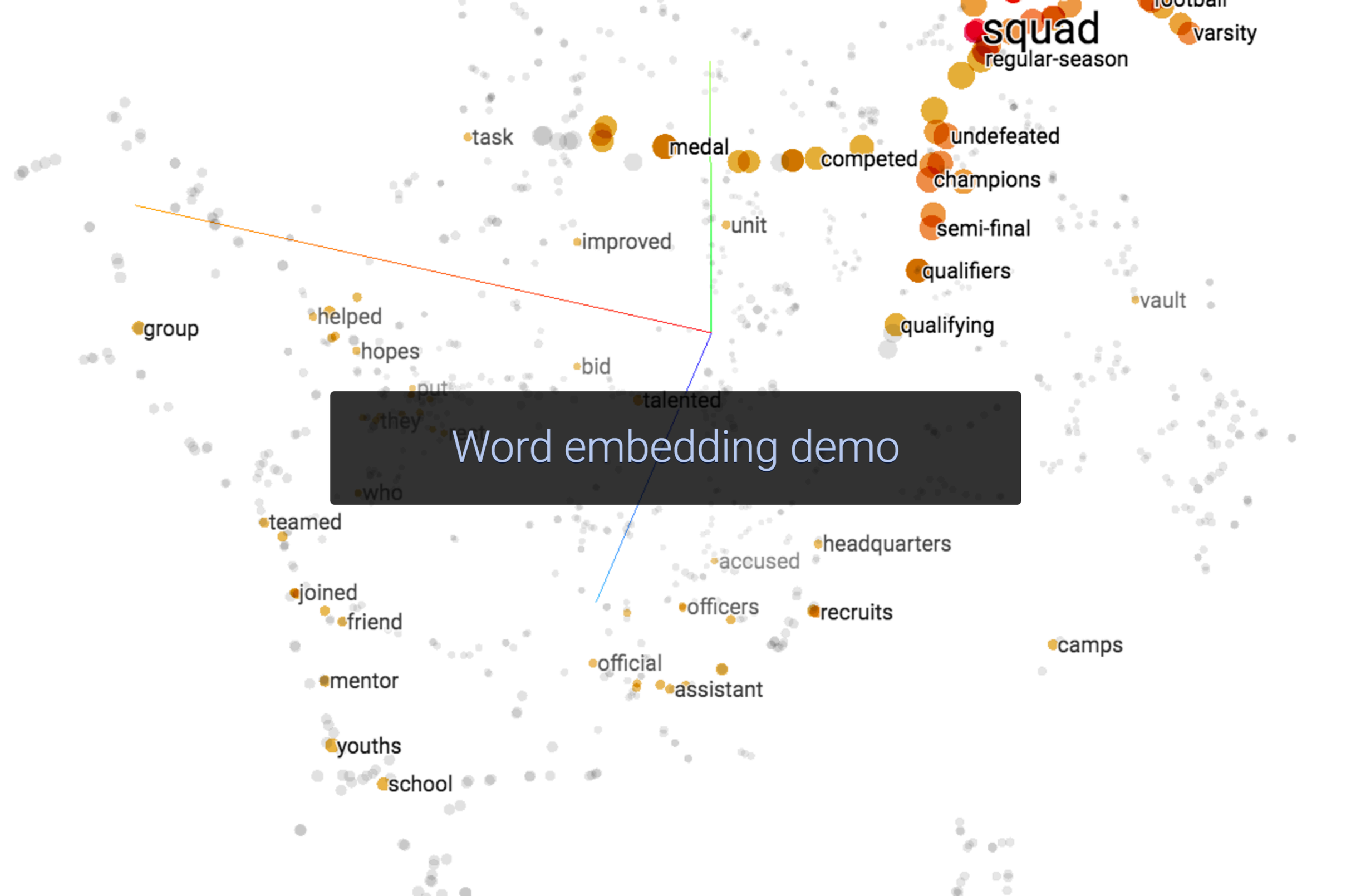
Word vectors

Represent words as high-dimensional vectors

Learn dimensions from co-occurrences in real text

Dimensions capture semantic relationships

Word vectors can learn biases from text


squad
regular-season
varsity
task
medal
competed
undefeated
champions
semi-final
qualifiers
qualifying
vault
unit
improved
group
helped
hopes
bid
put
talented
they
Word embedding demo
who
teamed
headquarters
accused
joined
friend
officers
recruits
camps
official
assistant
mentor
youths
school

A pattern language

A DSL to describe patterns over paths in the dependency tree

Describe root, as well as path from root to subject and object of relation.

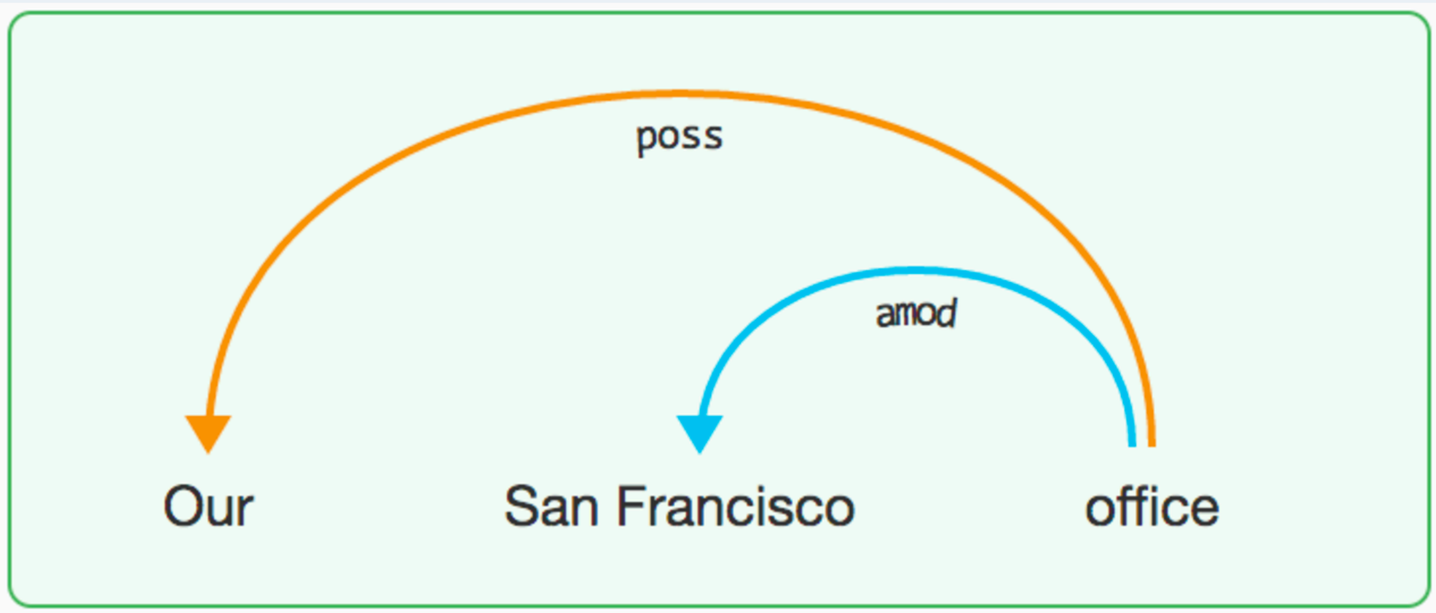
Can use word embeddings to create soft matches such as `office` and `located`

Can create reusable components such as `located_in_place`

```
# Possessive noun phrase
# eg "Our office in China..."
DepPath(
    head_to_start=[arc(poss, _)],
    head=token(office),
    head_to_end=located_in_place
)

# "in X", "located in X", etc
located_in_place = (
    maybe([arc(acl, located)]) + in_place |
    [arc(one_of(poss, compound, amod), _)] |
    [arc(amod, located), arc(npadvmod, _)]
)
```

Relation extraction demo



Thank you.